

Parcial de computabilidad

Lógica y computabilidad

Verano 2018

El examen es a libro abierto y se puede suponer demostrado lo dado en las clases y los ejercicios de las guías colocando referencias claras. Entregar cada ejercicio en hojas separadas. En cada hoja debe figurar nombre, apellido y número de orden.

Ejercicio 1. Considere la función $\text{listaOrdenada}(x) : \mathbb{N} \rightarrow \mathbb{N}$ que dada una lista x devuelve otra lista con todos los elementos de la lista x que son mayores que 0, ordenados de mayor a menor y sin repetir. Por ejemplo:

$$\begin{aligned}\text{listaOrdenada}([3, 1, 2]) &= [3, 2, 1] \\ \text{listaOrdenada}([3, 1, 2, 3, 4, 1]) &= [4, 3, 2, 1] \\ \text{listaOrdenada}([3, 1, 2, 0, 4, 1]) &= [4, 3, 2, 1]\end{aligned}$$

Demuestre que la función $\text{listaOrdenada}(x)$ es primitiva recursiva.

Ejercicio 2. Decida y demuestre si la siguiente función es computable.

$$g(x, y, z) = \begin{cases} 1 & \text{si } \exists w \text{ tal que } \Phi_x^{(1)}(w) \cdot \Phi_y^{(1)}(w) = z \\ \uparrow & \text{otro caso.} \end{cases}$$

Ejercicio 3. Decida y demuestre si la siguiente función es computable:

$$f(x) = \begin{cases} 1 & \text{si } \forall y \Phi_x^{(1)}(y) = y^2 + x \\ 0 & \text{otro caso.} \end{cases}$$

Ejercicio 4. Decida y justifique si los siguientes conjuntos son p. r., c. e., co-c. e. o computables:

- a. $C_1 = \{x : \forall y \Phi_x^{(1)}(2y) \uparrow\}$
- b. $C_2 = \{x : \forall y \Phi_x^{(1)}(2y) \downarrow\}$

1	2	3	4	N
A	A	A	A	10

1) Sea la función Lista Ordenada $(x): \mathbb{N} \rightarrow \mathbb{N}$ que devuelva una lista con los elementos de x mayores a cero, sin repeticiones y ordenados de mayor a menor, vemos que podemos definir las funciones:

* $seRepite(i, x) = \begin{cases} 1 & \text{si } x(i) \text{ se repite en } x \text{ al final} \\ 0 & \text{si no} \end{cases} = \begin{cases} 1 & \text{si } (\exists t \leq |x|) \ x(t) = x(i) \wedge t > i \\ 0 & \text{si no} \end{cases} = \alpha \left(\alpha \left(\min_{t \leq |x|} x(t) = x(i) \wedge t > i \wedge t \neq 0 \right) \right)$ (puesto que si \exists tal $t \Rightarrow x(t) = x(i)$ con $t > i$ y $x(i)$ se repite con índice mayor, y $t \in \{i+1, \dots, |x|\}$, $t > 0 \Rightarrow \alpha(x(t)) = 1$)

Vemos que $seRepite(i, x)$ indica si $x(i)$ se repite en x luego de i . Esta función es composición de la minimización acotada, α , operadores de lista y operadores lógicos. Todas ellas son p.r. \Rightarrow $seRepite$ es p.r.

* $sin Repes(x) = \text{lista } x \text{ sin elementos repetidos} = \min_{z \leq maxLista(x)} \underbrace{(\forall i \leq |x|) (\exists t \leq |z|) \ z(t) = x(i)}_{\text{todo elemento de } x \text{ está en } z} \wedge \underbrace{(\forall t \leq |z|) \alpha(seRepite(t, z))}_{z \text{ no contiene repetidos}}$
 * $maxLista(x) = nprimo(|x|+1)$

$sin Repes$ nos devuelve una lista con los elementos de x sin repeticiones (si alguno estuviera repetido, para la operación de menor índice de la repetición $\alpha(seRepite(t, z)) = 0$). Por otro lado, $maxLista$ nos da una cota superior para la minimización acotada de $sin Repes$, tomando el siguiente al último de la codificación de x y elevándolo al elemento máximo de x , que a su vez está elevado a la cantidad de elementos de x (la idea es tener una cota lo suficientemente grande para abarcar todos los datos necesarios).

Ambas funciones son composición de funciones p.r. por lo que $maxLista$ es p.r. y luego $sin Repes$ también.

En base a todas estas funciones podemos definir Lista Ordenada de x como:

* $Lista Ordenada(x) = \text{"lista de los elementos de } x > 0, \text{ sin repeticiones y ordenados de mayor a menor"} = \text{Lista Dec}(sin Repes(Maxores(x)))$

donde: $Maxores(x) = \min_{z \leq maxLista(x)} \underbrace{(\forall i \leq |x|) (\sum_{j=1}^{|z|} x(i) = z(j))}_{\text{todo elemento de } x \text{ aparece en } z \text{ igual cantidad de veces}} \wedge \underbrace{(\forall i \leq |x|) (\forall j \leq |z|) (x(i) > 0 \rightarrow \sum_{j=1}^{|z|} x(i) = z(j))}_{\text{todo elemento de } x > 0 \text{ aparece en } z \text{ igual cantidad de veces}} \wedge \underbrace{(\forall j \leq |z|) (\sum_{i=1}^{|z|} z(j) = z(j))}_{z \text{ no mayor a } x \text{ en longitud}}$

donde: $Maxores(x) = \min_{z \leq maxLista(x)} \underbrace{(\forall i \leq |x|) (x(i) > 0 \rightarrow \sum_{j=1}^{|z|} x(i) = z(j))}_{\text{todo elemento de } x > 0 \text{ aparece en } z \text{ igual cantidad de veces}} \wedge \underbrace{(\forall j \leq |z|) (\sum_{i=1}^{|z|} z(j) = z(j))}_{z \text{ no mayor a } x \text{ en longitud}}$

y * $Lista Dec(x) = \min_{z \leq maxLista(x)} \underbrace{(\forall i \leq |x|) (\exists t \leq |z|) \ z(t) = x(i)}_{\text{todo elemento de } x \text{ está en } z} \wedge \underbrace{|x| = |z|}_{\text{misma longitud}} \wedge \underbrace{(\forall j \leq |z|) \ z(j) \geq z(j+1)}_{\text{orden decreciente}}$

Es decir, $Maxores(x)$ devuelve una lista con los elementos de x mayores a cero (asegurándose de respetar su cantidad de apariciones en x) y

$Lista Dec$ nos devuelve los elementos de la lista x ordenados de forma decreciente, teniendo en cuenta que x es una lista sin elementos repetidos

(de no ser así la lista resultante podría contener elementos que no están en x). Ambas funciones son composición de funciones p.r. (la sumatoria es p.r.) por lo que ambas son p.r.

En base a su definición, Lista Ordenada de x nos devuelve la lista pedida y al ser composición de funciones p.r., esta también es p.r.

2) Sea $g(x, y, z) = \begin{cases} 1 & \text{si } \exists w \text{ tal que } \phi_x^{(1)}(w) \cdot \phi_y^{(1)}(w) = z \\ \uparrow & \text{si no} \end{cases}$ demostramos que es parcial-computable al exhibir el programa P:

(B)

P: $y \leftarrow y+1$ (tomamos $z_1 \leftarrow \langle w, t \rangle$ donde w es la variable definida en ϕ_x y ϕ_y , y t la cantidad de pasos de STP y SNAP)

[B] $z_2 \leftarrow \text{STP}^{(1)}(L(z_1), x_1, r(z_1))$ ($z_2 \leftarrow \text{STP}^{(1)}(w, x, t)$)

IF $z_2 = 0$ GOTO A (como $z_2 = 0$, el programa x no termina en t pasos para w , por lo que cambia la tupla)

$z_3 \leftarrow \text{STP}^{(1)}(L(z_1), x_2, r(z_1))$ ($z_3 \leftarrow \text{STP}^{(1)}(w, y, t)$)

IF $z_3 = 0$ GOTO A (como $z_3 = 0$, el programa y no termina en t pasos para w y cambiamos la tupla)

$z_2 \leftarrow \text{SNAP}^{(1)}(L(z_1), x_1, r(z_1))$ ($z_2 \leftarrow \phi_x^{(1)}(w)$)

$z_3 \leftarrow \text{SNAP}^{(1)}(L(z_1), x_2, r(z_1))$ ($z_3 \leftarrow \phi_y^{(1)}(w)$)

$z_4 \leftarrow z_2 \cdot z_3$ ($z_4 \leftarrow \phi_x^{(1)}(w) \cdot \phi_y^{(1)}(w)$)

IF $z_4 = x_3$ GOTO E (si $\phi_x^{(1)}(w) \cdot \phi_y^{(1)}(w) = z \Rightarrow$ el programa termina devolviendo 1)

[A] $z_1 \leftarrow z_1 + 1$ (z_1 para tener el siguiente valor de $\langle w, t \rangle$)

GOTO B (siguiente iteración del ciclo)

Como se puede ver, $\Psi_P(x, y, z) = \begin{cases} 1 & \text{si } \exists w \text{ tal que } \phi_x^{(1)}(w) \cdot \phi_y^{(1)}(w) = z \\ \uparrow & \text{si no} \end{cases} = g(x, y, z) \Rightarrow P \text{ computa } g \text{ y } g \text{ es parcial computable}$

Observación: los recursos utilizados en P son los de las funciones computables llamadas a funciones definidas en las guías prácticas.

podrías hacerlo en una

línea con un \exists

no es correcto.

3) $f(x) = \begin{cases} 1 & \text{si } \forall y \phi_x^{(1)}(y) = y^2 + x \\ 0 & \text{si no} \end{cases}$ Probaremos que esta función no es computable

Para ello, supondremos que lo es y tomaremos la función $g(x, y) = x + y^2$ la cual es p.r. por ser composición de funciones p.r. ($+$, x , a^b)

y por ende es computable. Con ella aplicamos el teorema de la recursión y vemos que existe un programa de número e tal que:

$\phi_e^{(1)}(y) = g(e, y) = e + y^2 \Rightarrow \underline{F(e) = 1}$. Usaremos esta idea para lo siguiente: Pero... ¿cuándo usas este e ?

Al definir $h(x, y) = \begin{cases} x + y^2 & \text{si } F(x) = 0 \\ \uparrow & \\ \text{si no} \end{cases}$ vemos que $h(x)$ es parcial computable pues F lo es (computable) y luego puede aplicarse al

teorema de la recursión para ver que $\exists e' \in \mathbb{N} / \phi_{e'}^{(1)}(y) = h(e', y) = \begin{cases} y^2 + e' & \text{si } F(e') = 0 \\ \uparrow & \\ \text{si no} \end{cases}$

Con esto, vemos que:

i) $F(e') = 1 \iff \forall y \phi_{e'}^{(1)}(y)(e') = y^2 + e' \text{ (definición de } F) \iff \forall y h(e', y) = y^2 + e' \text{ (} \phi_{e'}^{(1)}(y) = h(e', y) \text{)} \iff F(e') = 0 \text{ (definición de } h)$

ii) $F(e') = 0 \iff \exists y \phi_{e'}^{(1)}(y) \uparrow \vee \phi_{e'}^{(1)}(y) \neq y^2 + e' \text{ (definición de } F) \iff \exists y h(e', y) \uparrow \vee h(e', y) \neq y^2 + e' \text{ (} \phi_{e'}^{(1)}(y) = h(e', y) \text{)} \iff F(e') = 1$

(definición de h) En i) y ii) llegamos a un ABSURDO $\therefore F$ no es computable

(Alcuzaba solo con i) o con ii)

4) a) $C_1 = \{x: \forall y \phi_x^{(1)}(2y) \uparrow\}$

co-ce: si tomamos $\overline{C}_1 = \{x: \exists y \phi_x^{(1)}(2y) \downarrow\}$ luego vemos que la función característica de dicho conjunto es:

$g(x) = \begin{cases} 1 & \text{si } (\exists y) \phi_x^{(1)}(2y) \downarrow \\ 0 & \text{si no} \end{cases} = \alpha(\alpha(\min_{\langle t, w \rangle} STP^{(1)}(2tw, x, t)))$ (por minimización no acotada, si $(\exists \langle t, w \rangle) STP^{(1)}(2tw, x, t) \Rightarrow \alpha(\alpha(\langle t, w \rangle)) = 1$ y sino $g(x) = 0$) donde vemos que por ser composición de funciones parciales computables, g también lo es.

En consecuencia \overline{C}_1 es c.e. pues $\overline{C}_1 = \{x: g(x) \downarrow\}$ y luego C_1 es co-c.e. ✓

computable: Probaremos que C_1 no es computable suponiendo que lo es y tomando su función característica: $F_{C_1}(w) = \begin{cases} 1 & \text{si } \forall y \phi_w^{(1)}(2y) \uparrow \\ 0 & \text{si no} \end{cases}$

En base a eso, vemos que $C_1 = \{x: F(x)\}$ y C_1 es un conjunto de índices, puesto que si $x \in C_1$ y $\phi_x^{(1)} = \phi_{x'}^{(1)} \Rightarrow \forall y \phi_x^{(1)}(2y) \uparrow \Rightarrow$

$\Rightarrow x' \in C_1$. A su vez, vemos que dicho conjunto no es trivial, puesto que para los programas P y Q siguientes:

$P: \quad y \leftarrow y+1 \quad Q: \quad y \leftarrow y+1$ vemos que, si $\#P = e_1$ y $\#Q = e_2$, $e_1 \notin C_1$ ($\forall y \phi_{e_1}^{(1)}(y) \downarrow$) y $e_2 \in C_1$ ($\forall y \phi_{e_2}^{(1)}(y) \uparrow$) \Rightarrow
 $[A] \text{ IF } y \neq 0 \text{ GOTO } A \quad \Rightarrow \forall y \phi_{e_2}^{(1)}(2y) \uparrow$

En base a esto vemos que C_1 es un conjunto de índices no trivial, y aplicando el teorema de Rice, vemos que C_1 no es computable. ✓

ce: si C_1 fuese c.e., como es co-c.e. debería ser computable, sin embargo, no lo es \Rightarrow C_1 no es c.e. ✓

p.r.: Como C_1 no es computable, no es p.r. ✓

b) $C_2 = \{x: \forall y \phi_x^{(1)}(2y) \downarrow\}$ Probemos que el conjunto $Tot = \{x: \forall y \phi_x^{(1)}(y) \downarrow\}$ no es computable, p.r., ce ni co-c.e. Usamos si pode-

mos reducir Tot a C_2 . Queremos hallar $F: \mathbb{N} \rightarrow \mathbb{N}$ / $F(x) \in C_2 \Leftrightarrow x \in Tot$

$[F(x) \in C_2 \Leftrightarrow \forall y \phi_{F(x)}^{(1)}(2y) \downarrow \Leftrightarrow \forall y \phi_x^{(1)}(y) \downarrow \Leftrightarrow x \in Tot]$
 ↳ paso creativo

Para ello, tomamos $F(x) = s_1'(e, x)$ y vemos que $\forall y \phi_{F(x)}^{(1)}(2y) \downarrow \Leftrightarrow \forall y \phi_{s_1'(e, x)}^{(1)}(2y) \downarrow \Leftrightarrow \phi_e^{(1)}(2y, x) \downarrow \forall y \Leftrightarrow \phi_x^{(1)}(y) \downarrow \forall y$

y vemos que para $\phi_e^{(1)}(y, x) = \begin{cases} 1 & \text{si } (\exists t) (2xt = y \wedge \phi_x^{(1)}(t) \downarrow) \\ 0 & \text{si no} \end{cases}$ (2xt = y $\Leftrightarrow y = 2 \wedge x = t$)

* si $\forall y \phi_e^{(1)}(2y, x) \downarrow \Rightarrow \forall y (\exists t) 2xt = 2y \wedge \phi_x^{(1)}(t) \downarrow \Rightarrow \forall y \phi_x^{(1)}(y) \downarrow (2xy = 2y) \Rightarrow x \in Tot$

* si $\forall y \phi_x^{(1)}(y) \downarrow \Rightarrow \forall y \phi_e^{(1)}(2y, x) (2y = 2xy \wedge \phi_x^{(1)}(y) \downarrow) \Rightarrow$ por definición de $\phi_e^{(1)}$: $\phi_e^{(1)}(2y, x) \downarrow \Rightarrow \forall y \phi_{s_1'(e, x)}^{(1)}(2y) \downarrow$ (teorema del par)

mita: $\phi_{s_1'(e, x)}^{(1)}(y) = \phi_e^{(1)}(y, x) \Rightarrow \forall y \phi_{F(x)}^{(1)}(2y) \downarrow [F(x) = s_1'(e, x)] \Rightarrow F(x) \in C_2$

∴ $F(x) \in C_2 \Leftrightarrow x \in Tot \Rightarrow Tot \leq C_2$ y como Tot no es c.e., co-c.e., computable ni p.r., C_2 tampoco. ✓

Resultados:

	c.e.	co-c.e.	computable	p.r.
C_1	X	✓	X	X
C_2	X	X	X	X

